Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi

Vol. 9, No. 3, Juli 2018

ISSN 0216 – 0544
e-ISSN 2301– 6914

HYBRID INTRUSION DETECTION SYSTEM USING FUZZY LOGIC INFERENCE ENGINE FOR SQL INJECTION ATTACK

Rajif Agung Yunmar
Program Studi Teknik Informatika, Institut Teknologi Sumatera
Jl. Terusan Ryacudu Desa Way Huwi Kec. Jati Agung Kab. Lampung Selatan 35365
E-mail: rajif@if.itera.ac.id

Abstract

SQL injection attacks toward web application increasingly prevalent. Testing to the web that will published is the one of preventive measures. However, this method sometimes ineffective because constrained by various things. Instrusion detection system (IDS) has ability to help and protect the website from various attacks. This study proposed a hybrid IDS for web applications from SQL injection attacks. The IDS built based on hybrid architecture with a signature-based detection method, type of data that will be analyzed is network data packet and error log. The fuzzy logic inference engine used to be drawn the conclusion based on analyzed data. Proposed hybrid IDS tested against various types of SQL injection attack, such as Tautology, UNION query, Piggy-backed query, Malformed query, Stored Procedure, and Alternate Encoding. System testing is done with three scenarios with the aim of seeing the hybrid IDS response to the occurrence of false positives and false negative, including: testing with normal website access, testing with normal website access but inserting suspicious strings as part of SQL injection, and access that is truly a SQL injection attack. The result test shows that proposed hybrid IDS has good performance on detecting the various type of SQL injection attack, and significantly reduce or even remove the false positive and false negative.

Keywords: hybrid intrusion detection system, signature-based IDS, sql injection, fuzzy logic.

INTRODUCTION

Today the development of web-based applications growth rapidly. However, this is not followed by well knowledge from the web developers on important aspects of web security [1], its can caused the web applications vulnerable to attacks. Thousands of attacks attempted on various web applications around the world in a day [2]. SQL injection is one of the most common types of web attacking, and mostly dangerous [3].

There are several ways that can be done to protect the web from the attacking. Among them are testing the input type, checking the encoding input, and so forth [4]. However, this does not guarantee that the web application will be secure. Beside of the attacking techniques always growing, the web developer does not always know which parts of the web that has a weaknesses so it can exploited by the attacker.

Intrusion Detection System (IDS) is a software or hardware that performs to observation, inspection, processing, and correlates the information that collected then perform a particular action when evidence of the attack has been obtained [5]. This research proposed development of hybrid IDS with signature-based detection method approach. Detection of attacks based on data analysis of network data packet and web server error log. The uses of fuzzy logic inference engine to the two mentioned data is expected to improve detection accuracy and reduce the false positive and false negative.

INTRUSION DETECTION SYSTEM

Based on the architecture, an IDS can be classified into three types: host-based IDS, network-based IDS, and hybrid IDS. Where the host-based IDS works by analyze data from monitored hosts, network-based IDS analyze from network traffic, and hybrid IDS is the combination of both [6], [7].

Based on detection methods, an IDS can be classified into three types: signature-based, anomaly-based, and stateful protocol analysis [8]. Signature-based work by recognized attacks from previously known patterns. The working principle of this method in recognized an attack based on knowledge that have been defined before. So this method also called as knowledge-based detection or misuse detection. Anomaly-based detection works by observed the normal habits of the system. The anomaly will be known if there are something deviates from normal habit. For example: failed too many in login, excessive use of processor resources and memory, mass email delivery at a time, etc. This method is also called as behavior detection because of detection method that based on normal habits.

The workings of the stateful protocol analysis method is similar to anomaly-based detection, ie detecting possible anomaly on the system. The distinguishes is this method analyzes anomaly does not based on internal system or network standards, but based on standards that arranged by international standards organizations, for example from the Internet Engineering Task Force (IETF) organization. In general, the classification of IDS can be seen in Figure 1.

FUZZY LOGIC

Fuzzy logic is an inference engine that solved the problems with approaches like human reasoning. Fuzzy logic uses multilevel logic rather than true or false statements [9]. Fuzzy logic aims to formalize reasoning modes with an approximation rather than exact [10]. There are several reasons why the fuzzy logic uses, namely [11]:

1. The concept is easy to understand, it uses a simple mathematical concept.
2. Flexible and have a tolerance to inapposite data.
3. Can be modeled a complex nonlinear functions.
4. Can cooperate with conventional control techniques
5. Can build and apply expert experience without going through the training process.

Unlike classical logic which has only two values, fuzzy logic can have many membership values that are divided into degrees of membership and the degree of truth between 0 and 1 values. In fuzzy logic it is also possible that one value becomes partially true or partly wrong at the same time [12].

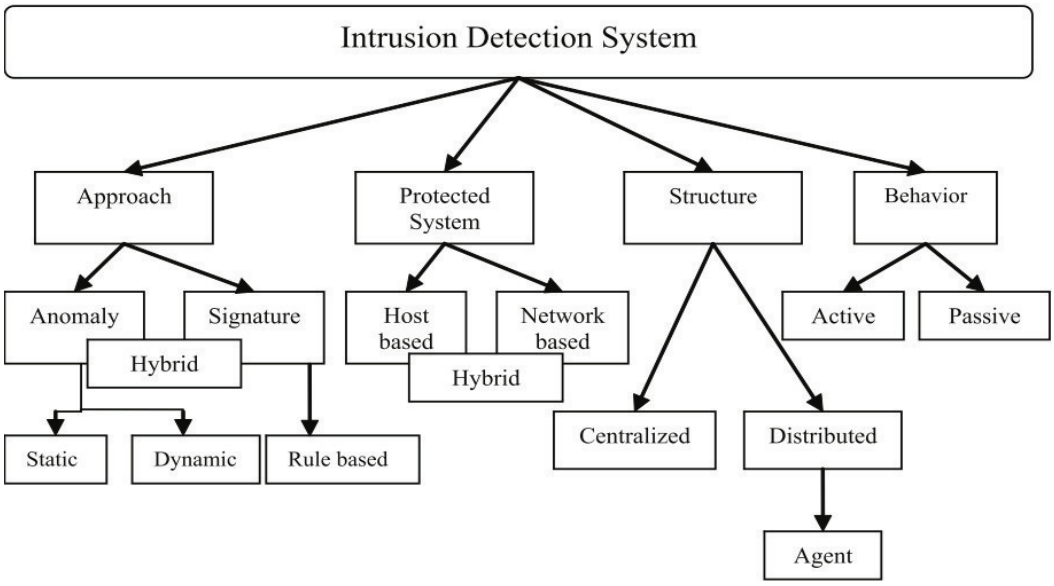

Figure 1. Classification of IDS [7]

SQL INJECTION

SQL injection is web attack that consists of characters or keywords entered by user to the parameters of web application that aimed to changing the meaning and logic of the actual SQL command [13]. SQL injection attacks caused the database can be accessed or manipulated by unauthorized people. A SQL injection attack can be classified into several groups, namely [14], [15]:

1. Tautology, SQL code injected into one ore more conditions, so can makes the SQL statement always true. Example:
2. `SELECT * FROM user WHERE name='admin' OR '1'='1' -- AND pwd='something'`
3. Union query, use the UNION keyword to get more data, UNION keyword injected through vulnerable parameters. Example:
4. `SELECT * FROM user WHERE name='admin' UNION SELECT * FROM balance -- AND pwd='something'`
5. Piggy-backed query, type of attack that uses query delimiter to inject additional SQL statements, Commonly has been known semicolon used as query delimiter. Example
6. `SELECT * FROM user WHERE name='admin'; DROP TABLE user -- AND pwd='something'`
7. Malformed query, the type of attack that exploits the error messages provided by the database from a SQL statement that is intentionally made wrong. Example:
8. `SELECT * FROM user WHERE name='admin' AND pwd='something' AND CONVERT(char, no)`
9. Stored procedure, types of attacks that utilize the standard stored procedure that is owned by common database. Example
10. `SELECT * FROM category where id=1; SHUTDOWN; -- ORDER BY name ASC`
11. Alternate encoding, type attacks that utilize special encoding alternatives, such as hexadecimal, ASCII, or Unicode. So, the injection codes can escape from the filter that applied by the web developer. The example of SQL statement:

    `1 UNION SELECT ALL FROM WHERE`

With alternate encoding, the SQL statement can be written as:

```
&#x31;&#x20;&#x55;&#x4E;&#x49;&#
x4F;&#x4E;&#x20;&#x53;&#x45;&#4C
;&#x45;&#x43;&#x54;&#x20;#x41;&#
x4C;&#x4C;&#x20;&#x46;&#x52;&#x4
F;&#x4D;&#x20;&#x57;&#x48;&#x45;
&#x52;&#x45;
```

```
[Thu Feb 01 10:30:11.940263 2018]
[mpm_winnt:notice] [pid 11956:tid
660]    AH00455:    Apache/2.4.10
(Win32)  OpenSSL/1.0.1i  PHP/5.6.3
configured  --  resuming  normal
operations

[Thu Feb 01 10:30:11.940263 2018]
[mpm_winnt:notice] [pid 11956:tid
660] AH00456: Apache Lounge VC11
Server   built:   Jul   17   2014
```

Figure 2. Example of web server error log

ERROR LOG WEB SERVER

Error log is the one of the most important log types. This is where the web server send the diagnostic information and record all of errors that occur while web application running. Error log is the first places to seen in case of a problem related to the operation of server [16]. Figure 2 shows an example of error log on the Apache web server associated with a query error that indicates a SQL injection attack has occurred.

RELATED WORKS

Several studies have been published in the topic of network security, especially in the field of intrusion detection system (IDS). The research classified into two categories, the first related with hybrid IDS, and two are associated with IDS for SQL injection attacks.

In first category, the studies generally built hybrid IDS with a combination of network data packet and system activity logs, among others Shanmugam and Idris [7], Sharma and Chandel [17], and Yunmar [18]. Except of Yunmar research, the first category generally focused on analyze network based attacks, such as DoS, U2R, R2L, etc.

While the research of hybrid IDS that discusses attack related with web application is not much.

In the second category, the development of IDS for SQL injection attacks built with signature-based methods. Where the data analyzed sourced from the network data packet that passing through port 80, as done by Maheswari and Anita [19], Kroné and Bahtijaragic [20], also Bhat and Mumbarkar [15]. While Irawan, et al. [14] developed a signature-based IDS with data sourced from port 3306.

The contribution of this research is looking for the impact usage of data source which in the previous research that very little even never to be used in the analysis of attack detection, that is combination of network data packet and web server error log. So, its expected to reduce or remove the false positive and false negative.

HYBRID INTRUSION DETECTION

In this study, the IDS built with hybrid architecture that combine network-based IDS and host-based IDS. Network-based IDS works at network level. Network-based IDS realized with the presence of Sniffer Agent in charge of capturing and analyze network data packet that captured through port 80. Meanwhile, host-based IDS works at application level.

Host-based IDS is realized with the presence of Log Analyzer Agent in charge of processing data that obtained from Apache error log. Suspicious things that found by Sniffer Agent and Log Analyzer will be sent to the Fuzzy inference engine to be determine the final result, whether classified into SQL injection attacks, or normal web access.

Figure 3 shows the proposed hybrid IDS design. This study uses signature-based approach as detection method. To recognize an attack, both of Sniffer Agent and Log Analyzer Agent works by extracting data with the Regular Expression techniques based on pattern that wrote in knowledge based.

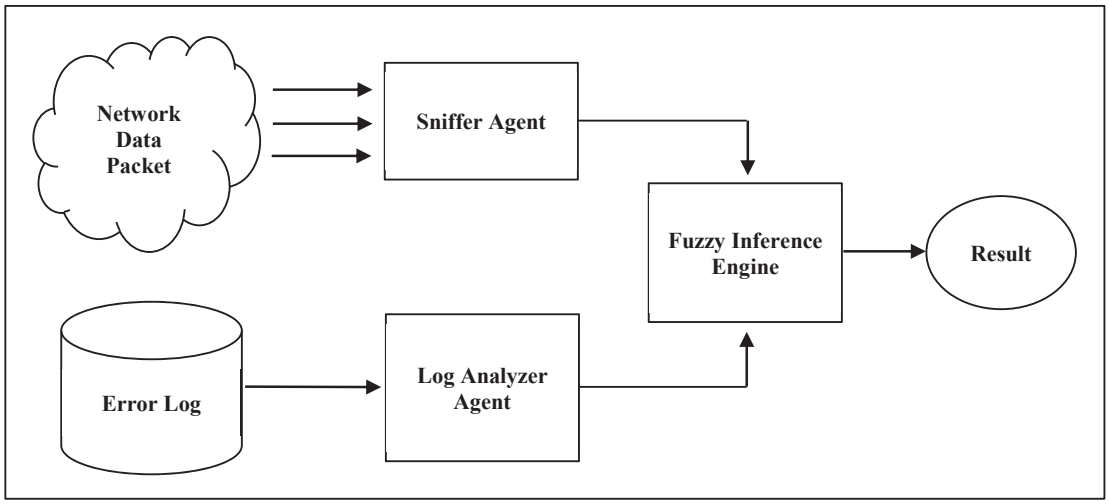

Figure 3. Proposed design of hybrid IDS

String that suspected contained injection

```
GET/?page=post-details&id=9+AND+1=2+
UNION+SELECT+0x6461726b3063306465,0x6
461726b3163306465,0x6461726b3263306465
,0x6461726b3363306465-- HTTP/1.1
Accept-Encoding: identity
Host: xyz.itera.ac.id
Connection: close
User-Agent: Python-urllib/2.5
```

Figure 4. Network data packet captured by Sniffer Agent that suspect as SQL injection attack

As the example shown in Figure 4, the Sniffer Agent captures the suspected data packet that indicates as an attack.

Sniffer Agent knows this is part of attacking attemp because the "UNION SELECT" string that contained on URI indicates as one of the SQL injection techniques. String that suspected by Sniffer Agent as an attack attemp is a string that associated with SQL keywords that just as described in Section 2.3. Among them are: UNION SELECT, SHUTDOWN, OR 1=1, DROP TABLE, and so forth. This attacking attemp will be reported to the Fuzzy inference engine to be determine the final result. The flow of attack detection process on Sniffer Agent can be seen in Figure 5.

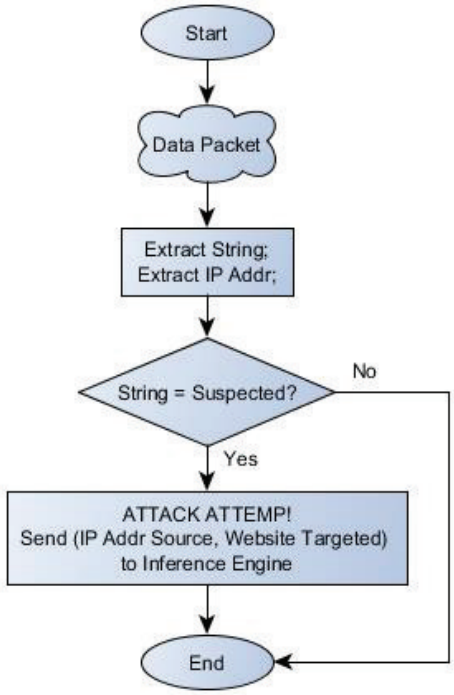

Figure 5. Sniffer Agent detect an attacking attemp

Log data that suspected as part of attack

```
Sat Nov 25 13:54:04.648118 2017]
[php5:error] [pid 25411] [client
116.206.42.118:59097] PHP Warning:
mysql_num_rows() expects parameter 1
to be resource, boolean given in
/rajiva/xyz/libs/my.lib.php on line
134
[Sat Nov 25 13:49:25.305181 2017]
[php5:error] [pid 25008] [client
116.206.42.118:59090] PHP Warning:
mysql_fetch_array() expects parameter
1 to be resource, boolean given in
/rajiva/xyz/libs/my.lib.php on line
189
```

Figure 6. Error log related with MySQL error

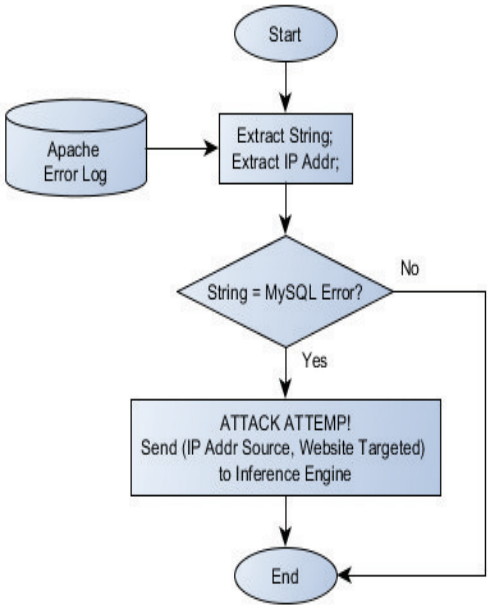

Figure 7. Log Analyzer Agent suspect the string is part of attacking attemp

Attack identification process in the Log Analyzer Agent applies the same principe as the Sniffer Agent done, its extract data that sourced from Apache error log with Regular Expression techniques based on the known pattern. As shown in Figure 6, Log Analyzer suspected those log item is part of attacking attempt because the error related with mistake in MySQL query, such as mysql_num_rows, and mysql_fetch_array. Log Analyzer will be reports this to the Fuzzy inference engine to be determine the final result. The flow of attack detection process on Log Analyzer Agent can be seen in Figure 7.

Every single attack attemp that recognized by Sniffer Agent and Log Analyzer Agent will be reported to the Fuzzy inference engine. The data that will be reported is IP Address of suspected attacker, and website targeted. The intensity of those attacking attemp can be input variables of the Fuzzy inference engine in determining the final decision, ie whether the activities performed by a user of an particular IP Address can be classified into SQL injection attacks or not.

INFERENCE ENGINE DESIGN

Linguistic Variable

Linguistic variable is the numeric intervals that have values, whose semantics are defined by their membership function. There are two variables that are used as consideration in performing an inference process, namely the error log variable and intensity variable sourced from network data packet that suspect as attacking attempt. Fuzzy logic inference engine will produce an output variable that indicate occurrence of attack activity.

First Variable

The 1st variable is intensity value of error log items appearence that sourced from a particular host or IP address. The graph of the membership function in this variable is described by the trapezoid as seen in Figure 8.

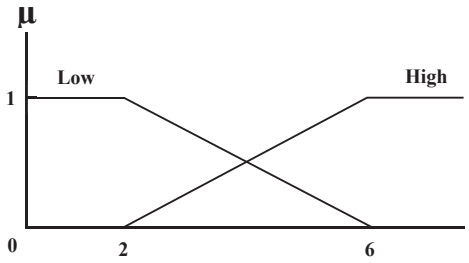

Figure 8. Membership function of 1st variable

Second Variable

The 2nd variable is intensity of network data packets that come from an IP address that suspected to be part of the attack. These values obtained from the Sniffer Agent which is responsible for processing the incoming network data packets. The graph of the membership function in this variable is shown as in Figure 9.

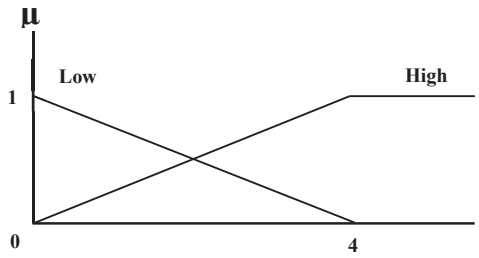

Figure 9. Membership function 2nd variable

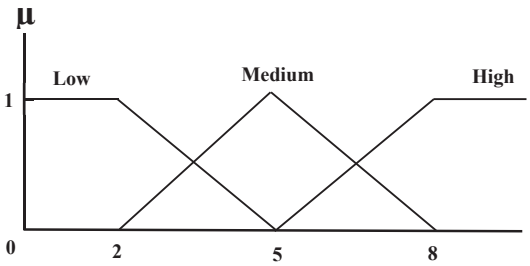

Figure 10. Membership function 3rd variable

Third Variable

The 3rd variable is the linguistic variable that determines the final output (defuzzification result) of proposed system which is indicates value the occurrence of an attack. The graph of membership function in this variable is illustrated in Figure 10.

Fuzzy Rule Base

The determination of the fuzzy rule base on the two linguistic input variables that mentioned before will be affect to the probabililty value of occurrence of attack. So, the fuzzy rule base can written as:

R1 = IF "Error Log Intensity" = "Low" AND "Intensity of Data Packet Suspected" = "Low" THEN "Attacking Indication" = "Low"

R2 = IF "Error Log Intensity" = "High" AND "Intensity of Data Packet Suspected" = "Low" THEN "Attacking Indication" = "Medium"

R3 = IF "Error Log Intensity" = "Low" AND "Intensity of Data Packet Suspected" = "High" THEN "Attacking Indication" = "Medium"

R4 = IF "Error Log Intensity" = "High" AND "Intensity of Data Packet Suspected" = "High" THEN "Attacking Indication" = "High"

SQL INJECTION SCENARIO

Attacking attempts begin when the attacker uses certain SQL injection techniques against the web through a computer network. This attacking activity creates a trace of log errors that produced by Apache web server. The Log Analizer Agent that recognized this pattern will reports it to the Fuzzy inference engine. The same thing also done by Sniffer Agent that recognizes the attack patterns of data packet that passing through a network.

Figure 11 illustrates an attack scenario until a hybrid IDS detected an attacking activity. SQL injection attacks can be done by manipulate the part of web that has direct access to queries, for example: URI, or input form.

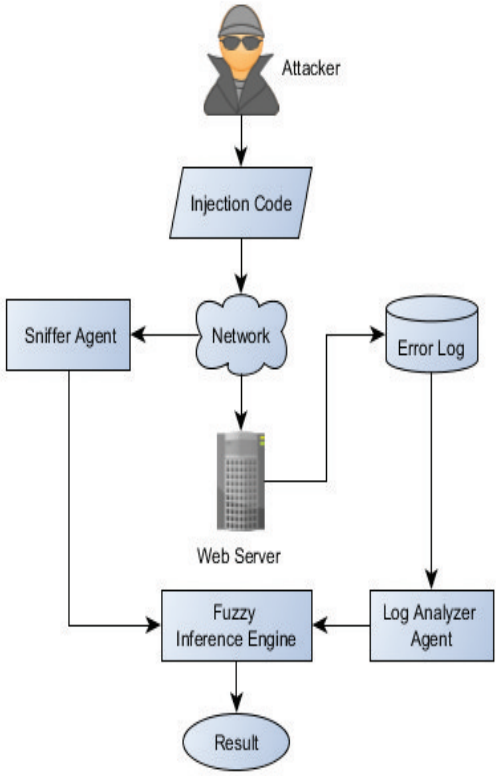

Figure 11. Attacking scenario

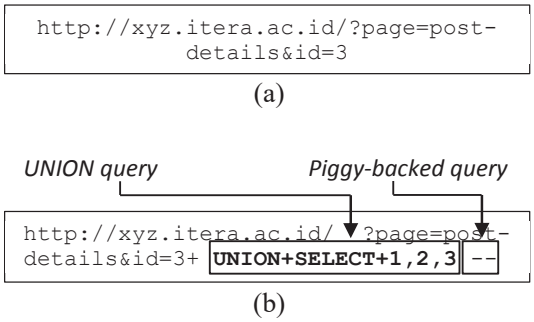

Figure 12. (a) Normal URI request, (b) URI request with UNION query attack

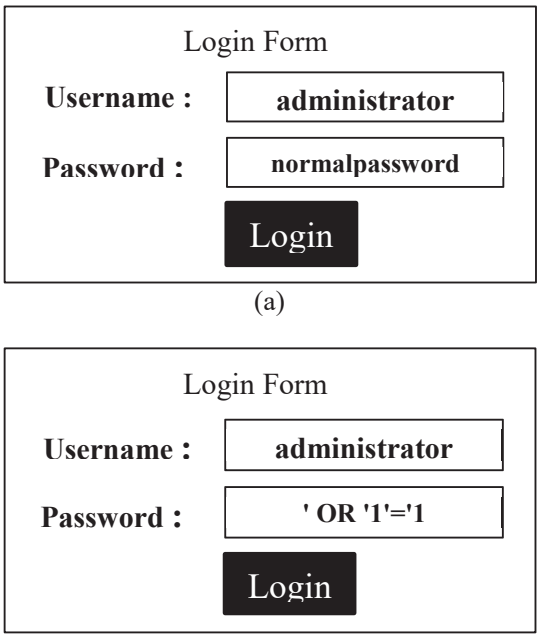

Figure 13. (a) Normal form login input, (b) Use of Tautology attack on the form login input

The SQL injection code can be manipulated in such a way as to be part of a normal URI, or a normal data that input to the form. For the example shown in Figure 12, that normal URI was manipulated by adding UNION query injection, and Piggy-backed query at once. Figure 13 shows how the input form (login) can be used as entry point for Tautology attack, one of SQL injection technique. This technique can be done by entering characters that are known to be used in Tautology-based attacks.

Generally, the tools that exist today such as Schemafuzz, SQL Map, and SQL Power Injector can replace the manual attacking as done above.

The working principle is the same, but this tools has the ability in automation, which is makes the tools can do the attacking attemp more faster than manual way.

SYSTEM AND DATA TESTING

The data used in this research comes from two sources, there is a network data packet, and an error log. Network data packet comes from scanning HTTP GET data packets that pass through Ethernet devices on port 80. While data of error log obtained from scanning performed on Apache error log. Various Linux distributions typically located that error at /var/log/apache/error_log. The error type of Apache error log that uses in study is MySQL error that produced during web application running. In this study the network data packet scanning task was handed over to the Sniffer Agent, while Apache error log scanning was handed over to the Log Analyzer Agent.

System testing conducted by using three scenarios, namely: testing with normal website access, testing with normal access but inserting data string pattern that recognized as part of SQL injection code, and testing that is truly a SQL injection attack. In each scenario, IDS will use several types of data to analyze. Types of data used include: network data packet only, Apache error log only, and a combination of network data packet and Apache error log. The purpose of this test is to see how far the IDS can distinguish which is really attack, and which is not, based on the type of data for analysis.

Table 1-3 shows the results of the tests conducted on the three proposed scenarios. While Table 4 shows the detail tests of several SQL injection type toward proposed hybrid IDS that uses combination of network data packet and Apache error log to analyze an attack.

Hybrid IDS written on Perl programming language that implemented on Linux-based operating system, Apache web server, and MySQL database. The testing of SQL injection attacks done by the various of penetration tools, such as Schemafuzz, SQLMap, SQL Power Injector. In addition, testing is also done

manually using web browser by writting the SQL injection command into web application. Table 5 shows the the testing detection results of various types of SQL injection attacks against the proposed hybrid IDS.

Table 1. IDS testing that only uses network data packet to analyze a web access

Scenario	Item Test	Number of Trial	Detected as SQL Injection Attack	Validity	False Positive	False Negative
1	Normal website access	120	0	100%	0%	**0%**
2	Normal access with inserted suspect SQL injection string	120	120	0%	100%	**0%**
3	Truly SQL injection test	120	120	100%	0%	**0%**
	Total	360	240	66.67%	33.33%	0%

Table 2. IDS testing that only uses Apache error log to analyze a web access

Scenario	Item Test	Number of Trial	Detected as SQL Injection Attack	Validity	False Positive	False Negative
1	Normal website access	120	0	100%	0%	**0%**
2	Normal access with inserted suspect SQL injection string	120	0	100%	0%	**0%**
3	Truly SQL injection test	120	95	79.17%	0%	20.83%
	Total	360	95	88.89%	0%	11.11%

Table 3. Hybrid IDS testing that uses combination of network data packet and Apache error log to analyze a web access

Scenario	Item Test	Number of Trial	Detected as SQL Injection Attack	Validity	False Positive	False Negative
1	Normal website access	120	0	100%	0%	**0%**
2	Normal access with inserted suspect SQL injection string	120	0	100%	0%	**0%**
3	Truly SQL injection test	120	120	100%	0%	**0%**
	Total	360	120	100 %	0%	0%

Table 4. Detail tests of hybrid IDS that uses combination of network data packet and Apache error log to analyze SQL injection attack

No.	SQL Injection Type	Number of Attempts	Detected as SQL Injection Attack	Validity	False Positive	False Negative
1	Tautologi	20	20	100%	0%	**0%**
2	Union query	20	20	100%	0%	**0%**
3	Piggy backed query	20	20	100%	0%	**0%**
4	Malformed query	20	20	100%	0%	**0%**
5	Stored procedure	20	20	100%	0%	**0%**
6	Alternate encoding	20	20	100%	0%	**0%**
	Total	120	120	100%	0%	0%

Table 5. Test result

No	Type of Attack	Tools	Result
1	Tautology	MN	Worked
2	UNION query	MN, SF, SM, SPI	Worked
3	Piggy-backed query	MN, SF, SM, SPI	Worked
4	Malformed query	MN, SF, SM, SPI	Worked
5	Stored procedure	MN, SF, SM, SPI	Worked
6	Alternate encoding	MN, SF, SM, SPI	Worked

Note: MN = Manually, SF = Schemafuzz, SM = SQLMap, SPI = SQL Power Injector

RESULT AND DISCUSSION

The IDS that uses only one data in analyzing an attack has potential to produces an error, both false positives or false negatives. The appearance of false positive is shown in Table 1, where something that is not an attack will be considered as a form of attack just because the data packets containing patterns that suspicious strings as part of SQL injection code.

The appearance of false negative can occur in IDS which only relies on data analysis from the Apache error log. Table 2 shows the existence of these error, namely the escape of a web access that should be identified as a SQL injection attack.

The value of false positives can be even greater if the attacker can write the injection code that can follow the SQL logic that used by programmers to develop the web.

Hybrid IDS that proposed in this study can eliminate errors as described above. The hybrid IDS uses two types of data to analyze the status of a web access. Hybrid IDS will cross-check a web access that indicated as an attack. For example, if the Sniffer Agent finds a string pattern of suspicious network data packet from an IP address, the hybrid IDS through the Log Analyzer Agent will cross-validate with the Apache error log data.

If the Log Analyzer Agent finds the appearance of MySQL error caused by web access from that IP address, then the IDS hybrid will conclude that user from those IP address is doing the trial of SQL injection attack.

The accuracy of proposed hybrid IDS can be calculated by Equation (1).

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP+FN}} \mathbf{x\ 100\%} \quad (1)$$

where TP represent True Positive, FP is False Positive, anda FN represent False negative. Testing result in quantitative shown in Table 6.

The development of IDS with a signature based method in many studies produces much of false positive and false negative. But in this study it can be reduce or even remove entirely by combining two data sources for material analysis, namely: network data packet and Apache error log. The performance comparation of proposed hybrid IDS and another studies shown in Figure 14.

Table 6. Test result in quantitative

Result	Percentage
True Positive	100%
False Positive	0
False Negative	0
Accuracy	100%

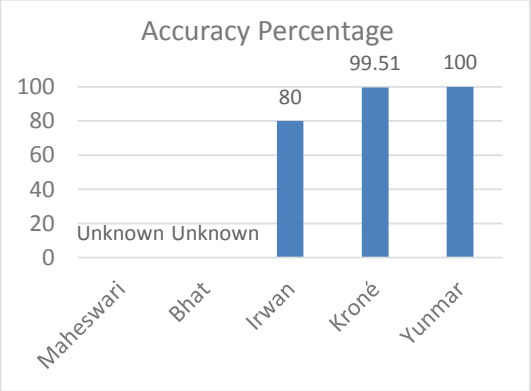

Figure 14. Comparation of studies

CONCLUSION

Hybrid IDS that proposed in this study has good result on detecting the various types of SQL injection attacks. Both of agents works based on a written pattern. An attack with a new pattern that did not wrote on knowledge base has great potential to escape from the detection system that applied by IDS. For the example: the attacking pattern code that not listed in the knowledge base will be not be detected by hybrid IDS agents.

In addition there are several types of web-based application attacks such as Cross Site Scripting (XSS), Local File Inclusion (LFI), Remote File Inclusion (RFI), and others that also require IDS handling. In the future, these issues are becoming challenge in research related with IDS development, especially IDS for web-based applications.

REFERENCES

[1] F. Valeur, D. Mutz, and G. Vigna, "A learning-based approach to the detection of SQL attacks," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, 2005, pp. 123–140.

[2] P. Technologies, "Web Application Attack Statistics Q1 2017," 2017.

[3] WhiteHat Security, "Web Applications Security Statistics Report 2016," 2016.

[4] M. P. Pathak, N. K. Khan, and T. C. Tantak, "A Survey to Detect and Prevent Web Attacks," *Int J Comput Sci Inf Technol Res*, vol. Vol. 4, No. 1, pp. 46–52, 2016.

[5] G. Vigna and C. Kruegel, "Host-Based Intrusion Detection," *Handb. Inf. Secur.*, pp. 1–35, 2005.

[6] Y. Lin, Y. Zhang, and Y. Ou, "The design and implementation of host-based intrusion detection system," in *Intelligent information technology and security informatics (iitsi), 2010 third international symposium on*, 2010, pp. 595–598.

[7] B. Shanmugam and N. B. Idris, "Hybrid intrusion detection systems (HIDS) using Fuzzy logic," in *Intrusion Detection Systems*, InTech, 2011.

[8] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.

[9] L. A. Zadeh, "Making computers think like people [fuzzy set theory]," *IEEE Spectr.*, vol. 21, no. 8, pp. 26–32, 1984.

[10] L. A. Zadeh, "Fuzzy logic systems: Origin, concepts, and trends," *Comput. Sci. Div. Dep. EECS UC Berkeley*, 2004.

[11] H. Purnomo and S. Kusumadewi, "Aplikasi logika Fuzzy untuk pendukung keputusan," *Yogyak. Graha Ilmu*, 2010.

[12] S. Kusumadewi, "Analisis dan desain sistem fuzzy menggunakan Toolbox Matlab," *Yogyak. Graha Ilmu*, 2002.

[13] C. Anley, "Advanced SQL injection in SQL server applications," 2002.

[14] A. S. Irawan, E. S. Pramukantoro, and A. Kusyanti, "Pengembangan Intrusion Detection System Terhadap SQL Injection Menggunakan Metode Learning Vector Quantization," *J Pengemb Teknol Inf Dan Ilmu Komput*, vol. Vol. 2, No. 6, pp. 2295–2301, 2018.

[15] A. Bhat and P. Mumbarkar, "Intrusion Detection And Prevention System: SQL-Injection Attacks," *Int J Sci Dev Res*, vol. Vol. 1, No. 5, pp. 18–21, 2010.

[16] "Log Files - Apache HTTP Server Version 2.4." [Online]. Available: https://httpd.apache.org/docs/2.4/logs.html#errorlog. [Accessed: 08-Nov-2017].

[17] R. K. Sharma and M. G. S. Chandel, "Novel Approach for Hybrid Intrusion Detection System," *Int. J. Adv. Res. Comput. Sci. Electron. Eng. IJARCSEE*, vol. 1, no. 9, p. pp: 43-47, 2012.

[18] R. A. Yunmar, "Intrusion Prevention System Untuk Aplikasi Berbasis Web," *None*, 2010.

[19] K. G. Maheswari and R. Anita, "An Intelligent Detection System for SQL Attacks on Web IDS in a Real-Time Application," in *Proceedings of the 3rd International Symposium on Big Data and Cloud Computing Challenges (ISBCC–16')*, 2016, pp. 93–99.

[20] M. Bahtijaragic and J. Kroné, "Intrusion Detection System by Statistical Learning," *LU-CS-EX 2016-26*, 2016.